NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A TESTBED TO EVALUATE COGNITIVE FEEDBACK THEORIES

by

Paul Kevin Larson

March 1991

Thesis Advisor: Kishore Sengupta

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/ AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b. DCLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b. OFFICE SYMBOL *(If Applicable)* Code 37	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c. ADDRESS *(city, state, and ZIP code)* Monterey, CA 93943-5000		7b. ADDRESS *(city, state, and ZIP code)* Monterey, CA 93943-5000	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	6b. OFFICE SYMBOL *(If Applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS *(city, state, and ZIP code)*		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
A TESTBED TO EVALUATE COGNITIVE FEEDBACK THEORIES

12. PERSONAL AUTHOR(S)
Larson, Paul K.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(year, month,day)*	15. PAGE COUNT
Master's Thesis	FROM TO	March 1991	58

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17. COSATI CODES			18. SUBJECT TERMS *(continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUBGROUP	Cognitive Feedback, Proces. Tracing, Information Boards, Cognitive Strategies

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Researchers have begun to use a variety of process tracing techniques to understand the cognitive processes underlying decision making and the effects of feedback on decision making. This study describes a computer program which monitors repetitive decision making behavior to allow researchers to infer the cognitive processes that underlie the use of decision feedback. This program has a number of commands which provide researchers with many ways to present decision data and feedback. The program uses a computer mouse and information board to implement process tracing based on the information acquisition search methodology. Brunswik's lens model is used as the decision making paradigm.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
[X] UNCLASSIFIED/UNLIMITED [ ] SAME AS RPT. [ ] DTIC USERS	Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Professor Kishore Sengupta	22b. TELEPHONE (Include Area Code) 408-646-3212	22c. OFFICE SYMBOL AS/SE

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted SECURITY CLASSIFICATION OF THIS PAGE
All other editions are obsolete Unclassified

Approved for public release; distribution is unlimited.

A Testbed to Evaluate Cognitive Feedback Theories

by

Paul Kevin Larson
Lieutenant, United States Coast Guard
B.S., United States Coast Guard Academy, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1991

Author: _____
Paul Kevin Larson

Approved by: _____
Kishore Sengupta, Thesis Advisor

_____
Tung X. Bui, Second Reader

_____
David R. Whipple, Chairman,
Department of Administrative Sciences

ABSTRACT

Researchers have begun to use a variety of process tracing techniques to understand the cognitive processes underlying decision making and the effects of feedback on decision making. This study describes a computer program which monitors repetitive decision making behavior to allow researchers to infer the cognitive processes that underlie the use of decision feedback. This program has a number of commands which provide researchers with many ways to present decision data and feedback. The program uses a computer mouse and information board to implement process tracing based on the information acquisition search methodology. Brunswik's lens model is used as the decision making paradigm.

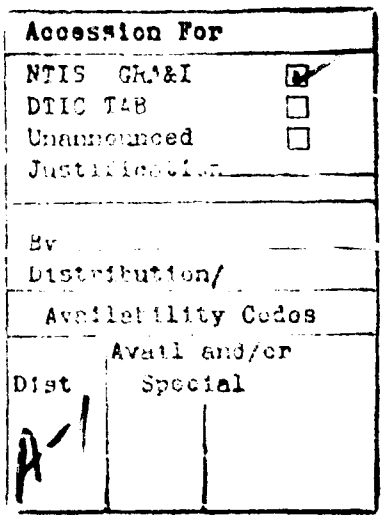

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND

Decision theory attempts to answer questions about human intelligence and behavior. How do people processes information, and how do they make decisions? Two techniques used to trace the decision making process are: verbal protocol analysis and information acquisition search. The first technique uses a subject who "thinks aloud" while making a decision. The verbal responses are analyzed to discern decision strategies. Research has shown that people have little insight into their decision processes, and they are often inconsistent in applying their own decision rules (Berry, 1987). Information that cannot be articulated by a decision maker is called implicit knowledge.

The second technique, information acquisition search, focuses on the data a subject collects when making a decision. An example is information boards. Flash cards containing data are tacked to a board. A subject turns the flash cards over, reads them, and then makes decisions. Which flash cards were examined, in what order, and for how long are recorded and analyzed. This method attempts to discover implicit knowledge to infer decision strategies, but it does not identify information a subject brings into the decision.

Cognitive feedback is a technique that gives a decision maker information about their own, and others, cognitive processes. Research has demonstrated the effectiveness of this technique as an aid for improving a decision maker's judgement rules. What elements of feedback are most effective, how do they

work, and what cognitive processes underlie the use of such feedback, have not been widely studied.

Cognitive feedback is given a firm theoretical foundation through the use of Brunswik's lens model of judgmental achievement. This is a structure that accents several important properties of decision making under uncertainty. The interaction among the decision maker and the environment is described in terms of several cues and their relationships. These relationships can be represented through statistical measures such as regression.

B. OBJECTIVES

This study had two major objectives. The first objective was to review decision and cognitive feedback theory and determine their theoretical basis, then use this information to identify areas which lend themselves to computer automation. The second objective was to develop a platform upon which to test cognitive feedback theories. Cognitive feedback has significant potential to raise the consistency and quality of repetitive decisions. This may be especially important when time pressures and stress are added to the decision making process. To this end, a computer program was written which used information acquisition search techniques to monitor the decision process.

C. RESEARCH QUESTIONS

This study focuses on two major questions. The first is: How to monitor the decision maker's information acquistion processes and the effects of feedback on them? A number of minor, related questions were also addressed:

- How to represent the decision making process?
- How to model the decision maker's cognitive processes?
- What is cognitive feedback?
- How to present cognitive feedback information?

The second question this study focuses on is: How to design and write a program to test and monitor the decision making processes? Minor, related questions are:

- How will such a program work?
- What is the program's specification?

D. SCOPE

One of the major objectives of this study is to define a testbed for cognitive feedback theory. There were two premises for this objective. The first premise was that the testbed would be a computer program that runs on a Sun Unix workstation. The second premise was that the software would be limited to collecting subject data in a "user friendly" manner. To this end, the software makes use of a graphical user interface (GUI). The testbed does not analyze the data collected.

The testbed focuses on decision making and feedback parameters. The scope of the decision making parameters include: task complexity and information availability. Task complexity refers to the amount of data given and the range of possible outcomes. Information availability would be constant, or on demand.

The scope of the feedback parameters include: content, feedback availability, and format. Content refers to the type of feedback, whether it is outcome feedback or one of many types of cognitive feedback. Feedback availability

refers to which types of feedback a subject receives and whether viewing the feedback is optional or required. Format refers to the appearance of the displayed feedback (e.g., bar-graphs or tables).

E. METHODOLOGY

This study began with a literature review of decision and cognitive feedback theories. Brunswik's lens model was chosen as the conceptual foundation for this study because it lends itself to computer automation. Original research was in the form of computer programming. Structured programming techniques were used in writing this program.

F. ORGANIZATION OF STUDY

This study contains four additional chapters. Chapter II discusses the theoretical framework of decision theory, Brunswik's lens model, feedback, and process tracing. Chapter III describes the program known as DEFT, *DEcision Feedback Testbed*. The chapter contains a general description of the program, detailed description of the program screens, and a discussion of the restrictions on the test data. Chapter IV describes the three types of files used by DEFT: text, control, and output. Chapter V summarizes the study and presents the conclusions that flowed from this research. The chapter concludes with a list of recommendations for future research.

II. THEORETICAL FRAMEWORK

Cognition is "the process by which the sensory input is transformed, reduced, elaborated, stored, recovered, and used" (Neisser, 1967). Cognitive psychology includes such areas as: memory, thinking, problem solving, perception, and language. This chapter discusses three areas of cognitive psychology: decision theory, cognitive feedback, and process tracing. It focuses on the decomposition of the judgment process after a judgment is made and the effects of providing the insights gained by this decomposition on decision makers. Cognitive feedback studies are concerned with how judgments are made, how to evaluate their quality, and ways to improve the judgment making process.

This chapter is divided into four sections. The first section discusses the general framework of decision theory by focusing on the lens model. According to Cooksey, Freebody, and Davidson, "the essential paradigm" of decision theory is embodied in the lens model (1986). The lens model gives decision theory a firm theoretical basis. The second section discusses cognitive feedback within the context of the lens model and identifies ways of displaying task information, the judgment maker's cognitive model, and the relationships between them. The third section discusses methods which allow researchers to study information acquisition and infer decision strategies. The final section is a short synopsis.

A. THE LENS MODEL

In most situations, decision makers cannot make judgments about an event or outcome based on direct observation. They make judgments based on

information which has some imperfect or statistical relationship to the outcome. An example is an employee hiring situation. Figure 1 shows a manager in this situation.

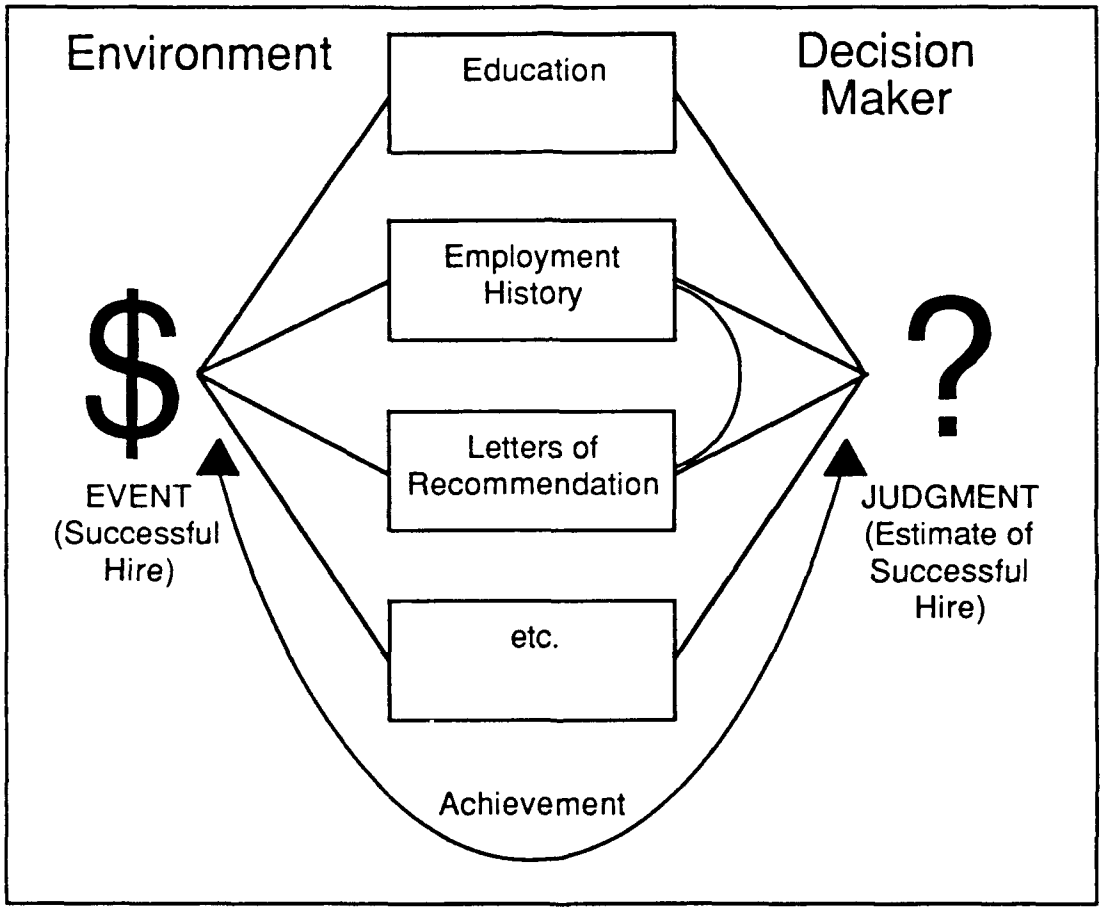

Figure 1. Sample Hiring Decision

The manager makes a judgment about the applicant through a "lens" of information. The pieces of information comprising the lens are known as cues. The cues reflect, to some degree, how well the applicant will perform the new job. The manager must weigh the available information to make a judgment.

Notice that relationships can exist between cues. These relationships are called correlations. The employment history of an applicant has a relationship to the letters of recommendation. Outstanding performance leads to outstanding letters of recommendation. The relationships between the manger and the cues, and the cues and the outcome, are described by the lens model. The lens model was proposed by Brunswick (1955) and it has undergone minor modifications. Figure 2 is the lens model as described by Dudycha and Naylor (1966). All the material presented here concerning the lens model is drawn from Libby (1981).

1. The Decision Maker

A manager (decision maker) tries to predict an applicant's suitability by looking at the education, employment history, letters of recommendation, etc. of an applicant. These pieces of information are called *cues* (X_i). Based on these cues, the decision maker forms a *judgment* (Y_S) on the applicant's suitability. The relationship between a cue and the judgement is uncertain, since the decision maker does not always incorporate cues in a consistent manner. Therefore, a statistical variable, *cue utilization* (r_{iS}), is used to describes the relationship between a single cue (X_i) and the decision maker's judgment (Y_S).

Given many judgments and a set of cues, it is possible to estimate the different cue utilizations using multiple linear regression. This process is known as a posteriori decomposition (Arkes and Hammond, 1986), and it implies that analysis is performed after a series of judgments have been made. Research shows that the accuracy of simple linear models is almost always approximately the same as the reliability of the judgments themselves and the introduction of more complex models rarely serves to significantly increase this accuracy (Goldberg, 1986).

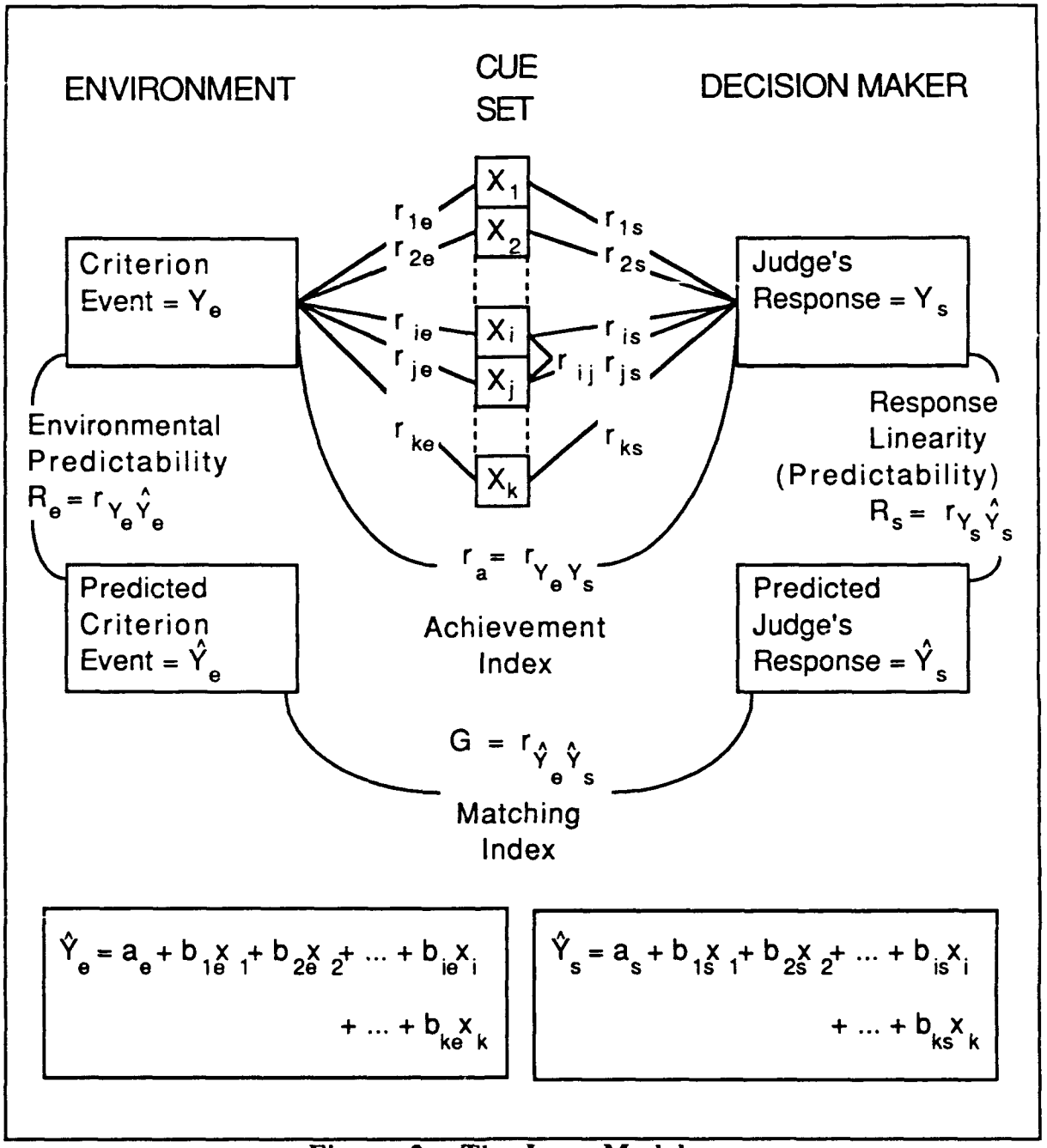

Figure 2. The Lens Model
Source: Dudycha and Naylor (1966, Figure 1)

There are five steps in computing cue utilizations. Step one is to compute the independent correlation matrix. Each element in this square matrix (I_{ij}) is the independent correlation of one cue (X_i) with another (X_j). Step two is to invert the independent correlation matrix. Step three is to compute the dependent correlation matrix. Each element in this column matrix (D_i) is the dependent correlation of cue (X_i) with judgment (Y_S). Step four is to compute the column matrix beta (β_S). Beta equals the matrix multiplication of the inverted independent correlation matrix and the dependent correlation matrix. The final step is to standardize the elements of beta (β_S), resulting in the cue utilization estimators (b_{1S}, b_{2S}, .., b_{kS}).

The *predicted judgment* ($\hat{Y}_S$) shows what the decision maker's judgment would be if the cue utilization estimators were consistently applied. This value can be computed by: $\hat{Y}_S = a_S + (b_{1S} \cdot X_1) + (b_{2S} \cdot X_2) + ... + (b_{kS} \cdot X_k)$. Where a_S is a constant adjustment factor which makes the result fall within the judgmental range. The *response linearity* ($R_S = r_{Y_S \hat{Y}_S}$) is the relationship between judgment (Y_S) and the predicted judgment ($\hat{Y}_S$).

2. The Environment

Decision makers attempt to predict an outcome based on available cues. The actual outcome, or event, is called the *criterion* (Y_e). Most cues are not perfect predictors of a criterion. Some cues may be irrelevant, other cues may overlap or duplicate other cues, and important information may be missing. The decision maker operates within an uncertain environmental context. Therefore, a statistical variable, *ecological validity* (r_{ie}), is used to describes the relationship between a single cue (X_i) and the criterion (Y_e).

In the same manner that cue utilization is estimated, ecological validity can also be estimated. The steps differ only by substituting the criterion (Y_e) for the judgment (Y_s). The resulting matrix beta is denoted by β_e, and the ecological validity estimators are b_{1e}, b_{2e}, .., b_{ke}.

The *predicted criterion* ($\hat{Y}_e$) shows what the outcome would be if the environment could be perfectly predicted. This value can be computed using: $\hat{Y}_e = a_e + (b_{1e} \cdot X_1) + (b_{2e} \cdot X_2) + ... + (b_{ke} \cdot X_k)$. Where a_e is a constant adjustment factor which makes the result fall within the range of possible outcomes. The *environmental predictability* ($R_e = r_{Y_e \hat{Y}_e}$) is the relationship between criterion (Y_e) and the predicted criterion ($\hat{Y}_e$).

3. Relationships Between Environment and Decision Maker

The *achievement index* ($r_a = r_{Y_e Y_s}$) measures the decision maker's performance at predicting the outcome. The *matching index* ($G = r_{\hat{Y}_e \hat{Y}_s}$) measures a consistent decision maker's performance in a consistent environment. Since the environment is not consistent, this reflects the maximum performance that a given decision making strategy can achieve.

B. FEEDBACK

Feedback (FB) is defined as the return to the originating system of some evaluative information about a task. The purpose of feedback is to allow the originating system to adjust itself based on this information, and to bring itself closer to the ideal state (Doherty and Balzer, 1988). According to Doherty and Balzer, there are two major types of feedback: outcome feedback (OFB) and cognitive feedback (CFB).

1. Outcome Feedback

Outcome feedback (OFB) simply presents decision makers with the information about the actual results of a judgment by giving them the criterion (Y_e). Studies have shown that OFB can be dysfunctional. According to Arkes and Hammond:

> A wholly fortuitous discovery by Newton (1965), however, that subjects might well be able to improve their performance without outcome feedback led Todd and Hammond (1965) to investigate an alternative type of feedback. They showed that if subjects were given feedback of a cognitive nature (that is, information about the properties of task systems and their judgment systems), they could rapidly improve their performance without outcome feedback (that is, without being told the correct answer after each trial). Moreover, they found that providing outcome feedback in addition to cognitive feedback did not improve accuracy. Indeed, Hammond, Summers, and Deane (1973) later showed that adding outcome feedback could result in the impairment of performance. (1986, p. 66)

The problem with OFB is that it does not show the decision maker where or how they can improve their performance.

2. Cognitive Feedback

The second major type of feedback is cognitive feedback (CFB). CFB presents decision makers with information about how or why a judgment was made. It provides information about their own judgments, the environment, and the applicability of each cue to the criterion. Cognitive feedback contains three distinct types of information: task information (TI), cognitive information (CI), and functional validity information (FVI).

TI refers to information about the environment. Where environments are inferred by their effects on the relationships between the cues and the

criterion. These relationships are: environmental predictability (R_e), cue ecological validity (r_{ie}), and cue intercorrelation (r_{ij}) (Balzer, Doherty, and O'Conner, 1989).

CI refers to information about the decision maker, where the decision makers' cognitive models are inferred from the judgments they made based on the cues. These relationships are: response linearity or predictability (R_S), and cue utilization (r_{iS}) (Balzer et al, 1989).

The final type of information contained in cognitive feedback is FVI. FVI refers to information about the environment criterion and the decision maker's judgments. These relations are: achievement index (r_a), matching index (G), and the correlation between the residuals from the predictions (C) (Balzer et al, 1989).

Each type of CFB information can be presented in many ways. Experiments have been conducted testing the effects of: function forms, graphs, bar-graphs, tables, weights, and verbal reports. In addition, different types of CFB information can be combined and shown to decision makers at the same time. For a summary of these experiments, see Balzer et al (1989).

C. PROCESS TRACING

Early cognitive process studies adopted a "black box" view. They presented a subject with a task and studied the output. Decision researchers have become interested in trying to get "inside" these processes and explain how people make decisions. These "white box" studies are called *process tracing*. According to Johnson, Payne, Schkade, and Bettman (1989), there are two process tracing techniques of major interest to researchers: verbal protocol analysis and information acquisition search.

1. Verbal Protocol Analysis

The first technique uses a subject who "thinks aloud" while making decisions. The subject's verbal responses are then analyzed to discern decision strategies. This technique has two major benefits. The first is that it works in numerous situations. According to Ericson and Simon (1984), verbal protocol analysis has proven it can provide useful data in the study of cognitive processes. The second benefit is that a subject can discuss additional process data. Additional process data is information that a subject brings into the decision.

There are a number of problems with verbal protocol analysis. When asked how they steer a bicycle, most people believe they turn the handlebars to steer. Physics has proven that this is true only at slow speeds. At normal and high speeds people shift their weight, which applies torque to the spinning wheels, and causes a change of direction. Most people ride bicycles without comprehending how they perform a basic function. Information that people have which they cannot articulate is called *implicit knowledge*. Similarly, research has shown that people often have little insight into their decision processes, and they are often inconsistent in applying their own decision rules (Berry, 1987). Another problem with verbal protocol analysis is that it is generally not good at getting deep casual knowledge about the procedures or semantics of a system and it is limited to information that people can articulate (Olson and Rueter, 1987). The third problem is that the process of making verbal reports becomes a secondary task which may affect the subject's performance of the task under study. Finally, verbal reports are difficult to analyze formally (Russo, 1978).

2. Information Acquisition Search

The second process tracing technique, information acquisition search, focuses on the data a subject examines, in what order, and for what interval. This technique attempts to discover implicit knowledge to infer decision strategies. According to Payne, Braunstein, and Carroll (1978), the information acquisition technique has been successfully used to study the decision process. The simple information board with flash cards, mentioned in Chapter I, is an example of an information acquisition search technique. Other examples include a system which records eye movement (Russo, 1978) and a system which records the motion of a computer mouse (Johnson et al, 1989).

People generally acquire information in three to four seconds using simple flash cards and in 200 to 300 milliseconds using eye movement (Russo, 1978). Ideally, monitoring eye movement is the best way to collect information acquisition search data, but a system which collects this data is "quite expensive" (Russo, 1978). A low cost alternative can be found in the computer mouse. While not ideal, the computer mouse is fast, easy to use, and inexpensive. According to Johnson et al:

> An analysis of the time to move the mouse from point to point suggests that this follows Fitts Law (Card, Moran, & Newell, 1983). They suggest that the time to move a mouse is primarily limited by the central information-processing capacities of the eye-hand guidance system. In other words, the major limitation in speed is due to the time it takes to think where to point, not in the movement of the mouse. Further analysis of the performance of the mouse indicated that it was within 5% of this optimal pointing device described by Fitts Law. (1989, p. 4)

A disadvantage of the information acquisition search technique is that it cannot identify information a subject brings into the decision, it can only identify information that a subject collects.

D. SYNOPSIS

Within the Department of Defense, as in most professions, there is a constant need to improve decision making in repetitive situations. Decision researchers have described many new methods to study and improve decision making behavior and Brunswik's lens model is the central paradigm for many of these methods. The remainder of this study describes a computer program designed to test cognitive feedback theories. This program uses information acquisition search techniques to infer decision strategies and monitor the effectiveness of the feedback. This program is an information board which uses the computer mouse to perform the decision task.

III. DECISION FEEDBACK TESTBED

A. GENERAL DESCRIPTION

This chapter describes the program known as DEFT, *DEcision Feedback Testbed.* This program interacts with an experiment subject (user) to test the effects of cognitive feedback on user training and performance. DEFT is written in the C programming language for the Sun Unix workstation. DEFT's user interface development was accelerated through the use of the SunView™ graphical user interface (GUI) toolkit. While admittedly subjective, using SunView interface items increased the ease of use and understandability of the DEFT program.

There are two general ways in which a person interacts with a computer (Hartson and Hix, 1989). The first way is a sequential dialogue. The user moves sequentially from one dialogue to another. A hierarchy of menus and screens is a common trait. It is generally easier to program using this method and it is appropriate when the task itself is sequential (Powers, Cheney, and Crow, 1990). The second way a person interacts with a computer is direct manipulation. The user "grabs" items and manipulates them. Keyboard input is replaced by a pointer and mouse. Direct manipulation is easier to learn and use (Hartson and Hix, 1988). DEFT uses a combination of both techniques. The program moves sequentially through start up, block, and feedback screens. This follows logically from the task, but once a screen is displayed, there are many direct manipulation screen items. Direct manipulation is more intuitive and allows the user to be "spontaneous" in choosing what to do next. Direct

manipulation items in DEFT include: the pointer, three button mouse, scrollbars, and text insertion point. A short training session in these areas should be given to novices.

It is the *task* of the user to provide a judgment for a set of cues. DEFT groups a number of tasks within a *block*. DEFT uses a simple information board, based on the spreadsheet metaphor, to present the block to the user. Each task is a separate row and the cues are aligned in columns. The first column is reserved for the user's judgment (response). DEFT is very flexible when it comes to the handling the surface characteristics of cues. DEFT sets no limit on the number of cues provided to the user since it uses dynamic data structures to store cues. The experimenter may wish to limit the number of cues, since only five cues are visible without scrolling the window horizontally. DEFT also sets no limit on the number of tasks. Although only 20 tasks are visible without scrolling the window vertically. DEFT does not limit cues to numeric values. It supports three metric characteristics: numeric, discrete, and categorical which can be represented by strings of 14 characters.

DEFT uses a linear regression model, based on the lens model described in Chapter II, to analyze the user's judgments. This model is limited to simple cognitive systems. According to Libby:

> ...a regression (or discriminant analysis) model, constructed by regressing the criterion event on the available cues, was shown to outperform human judges. Unlike the judge, these models make perfectly reliable predictions and optimally weight the available cues (in the least-squares sense). A fairly stringent set of environmental conditions is necessary for construction of this type of model. Quantitative specification of both the decision-relevant cues and the criterion event is required, as is a sufficient number of cases to estimate the parameters of the model. These conditions limit applications to a subset of repetitive decisions. (1981, p. 105)

The user can receive feedback after entering a series of task judgments. Based on these judgments, posteriori decomposition ("after the fact" analysis) reveals the user's cue utilizations. DEFT uses a double system paradigm and the experimenter must provide the criterion events (target responses). Without criterion events, DEFT feedback would be limited to cognitive information (CI) and could not provide outcome feedback (OFB), task information (TI), or functional validity information (FVI).

This is the first version of DEFT and it incorporates four forms of feedback. The first form of feedback is target responses (OFB). The second form of feedback is target cue weights in bar-graph format (TI). The third form of feedback is user cue weights in bar-graph format (CI). The fourth form of feedback is consistent user responses (CI).

B. THE START UP SCREEN

DEFT requires that the user identify himself and the experiment. At start up, DEFT displays the screen shown in Figure 3. The user identifies himself by providing the following personal information: first name, last name, student mailbox code (SMC), age, and sex. This information is used for two purposes. The first purpose is to separate or identify, experiment output files from multiple users. The second purpose is to collect demographic information to identify group trends and statistics.

The user identifies the experiment he wishes to run by entering an Experiment File name. The experimenter must provide this file name to the user. The Experiment File contains a script which controls the experiment. DEFT reads this file, line by line, and it is the method by which an experimenter interacts with the user.

Figure 3. Start Up Screen

Input into the screen fields is via the keyboard, while function selection is controlled by manipulating the pointer and clicking on one of three screen function buttons. Two buttons are self explanatory; the Help button provides information on using SunView, and the Quit button terminates the program. If either button is "pressed", the contents of the user input fields are ignored. When the user presses the Next button, the contents of the user input fields are validated. Validation consists of ensuring that the personal data fields are not empty, and that the Experiment File exists. After validation, DEFT interprets the Experiment File to determine what happens next.

C. HELP

DEFT has a simple help facility used throughout the program. The user requests help by pressing the Help button. The help text cannot be changed within DEFT and the help given to the user is not context sensitive. The help text is shown to the user within a read only, scrolling window. The user scrolls the text by direct manipulation of the scrollbars with a pointer and mouse. The user signals he has read the help text by pressing the Continue button. The help facility was added to DEFT as a mechanism for the user to get generic information about the SunView environment. Information unique to an experiment should not be given with help, but should use the instruction facility. The help text is not stored in the DEFT program, but exists under Unix as an ASCII text file named "deft.help". Outside of DEFT, this file can be edited by the experimenter.

D. INSTRUCTIONS

DEFT has an instruction facility. Instructions are shown when directed by the experimenter. The experimenter controls their display through the Experiment File. When an instruction command is encountered within the Experiment File, the user is shown the contents of an instruction text file within a read only, scrolling window. The user signals he has read the instructions by pressing the Continue button. The instruction facility was added to provide context sensitive help to the user as he moves through an experiment. Throughout the experiment, the user can review the last set of instructions by pressing the Instruction button.

E. QUESTIONNAIRES

DEFT also has an questionnaire facility. Questionnaires are given to the user when directed by the Experiment File. When a questionnaire command is encountered within the Experiment File, the questionnaire is copied and the user is given the copy to answer. The questionnaire appears within an modifiable, scrolling text window. The user signals that he has completed the questionnaire by pressing the Finished button. Questionnaires can be an efficient way of gathering subjective information about the experiment. They can gather information about the relations, objects, inference rules, and feelings of the user. Completed questionnaire are stored under a file name that consists of the questionnaire file name concatenated with the user's last name.

F. THE BLOCK SCREEN

DEFT requires that the user perform a series of tasks (a block) before it can provide the user with feedback information. A sample block screen is shown in Figure 4, and it consists of three windows. The data window is on the bottom, and it is the largest of the windows. It is a simple information board that uses a spreadsheet metaphor to present a series of tasks to the user. Each task is a separate row. The first column in a row is the judgment input field. The other columns are the cues for that task. The small window, in the upper left corner, shows the set of possible user responses. The third window, in the upper right corner, is the control panel.

The control panel consists of buttons and an optional digital timer. There are four buttons which always appear in the control panel. First, the Help button which provides the user with information about SunView. Second, the Instruction button which allows the user to review the latest instructions. Third, the Next button which the user presses when he has completed the block and reviewed the feedback. When the user presses the Next button, the contents of the user input fields are validated. Validation ensures that each judgment is within the set of possible responses. After validation, DEFT interprets the Experiment File to determine what happens next. Fourth, the Quit button which terminates the program.

In addition, there are four optional buttons corresponding to the four types of feedback. The experimenter controls the display of feedback through the Experiment File. These buttons only appear if the associated feedback is enabled.

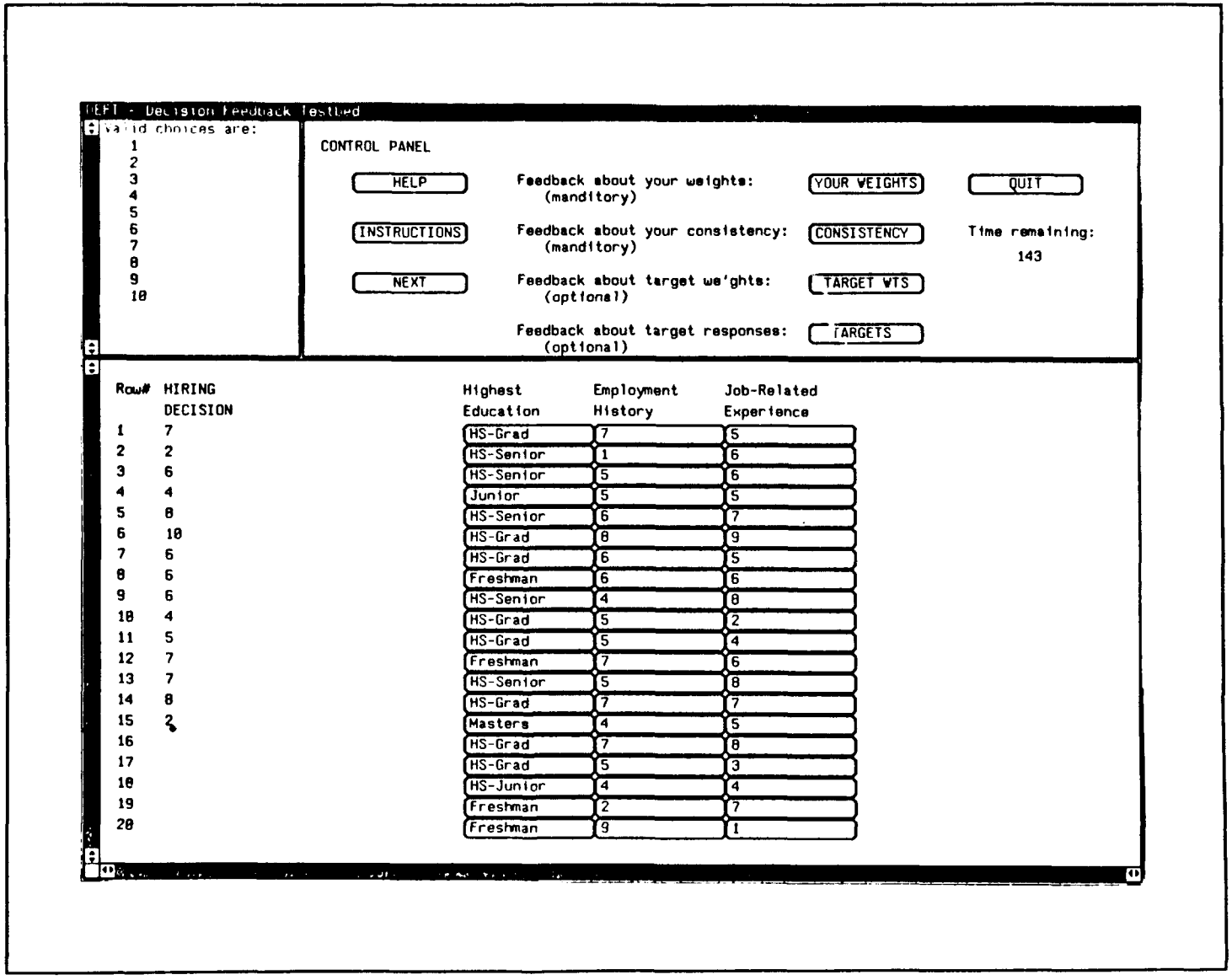

Figure 4. Sample Block Screen

The final control panel item is the optional digital timer. The timer shows the number of seconds remaining for the user to enter judgments. If the timer is enabled, it appears under the Quit button. The timer is set within the Experiment File.

G. FEEDBACK

The user can receive feedback after entering a minimum number of task judgments. DEFT provides four forms of feedback: outcome (OFB), task information (TI), and two types of cognitive information (CI). Only one form of feedback can be shown at a time and the experimenter controls the availability of feedback through the Experiment File.

Two forms of feedback are shown on the block screen: OFB and CI. OFB is given to the user by showing him the criterion event (target response) for each task for which he has made a judgment. These responses appear under a new column in the spreadsheet next to the user response input field. Once the user has viewed the criterion event for a task, the associated user response input field is locked and the user cannot change it. Figure 5 shows a sample target response feedback screen.

In a similar manner, CI is given to the user by showing him the predicted judgments (consistent user responses) and allowing him to compare them with his judgments. This provides the user with a measure of his response linearity. Unlike OFB, the user response input field is not locked and the user can change his responses. Figure 6 shows a sample consistent user responses feedback screen.

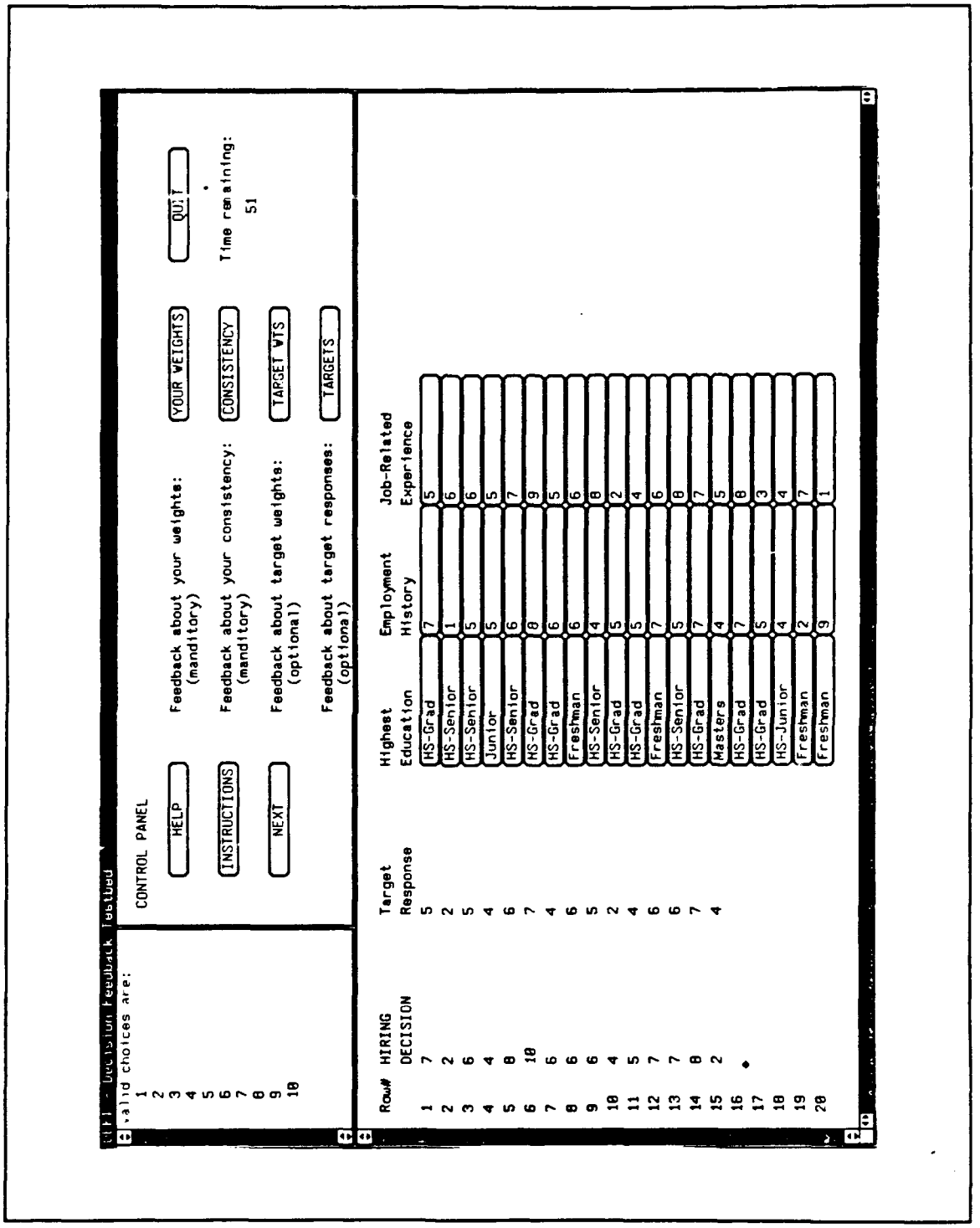

Figure 5. Sample Target Response Feedback Screen

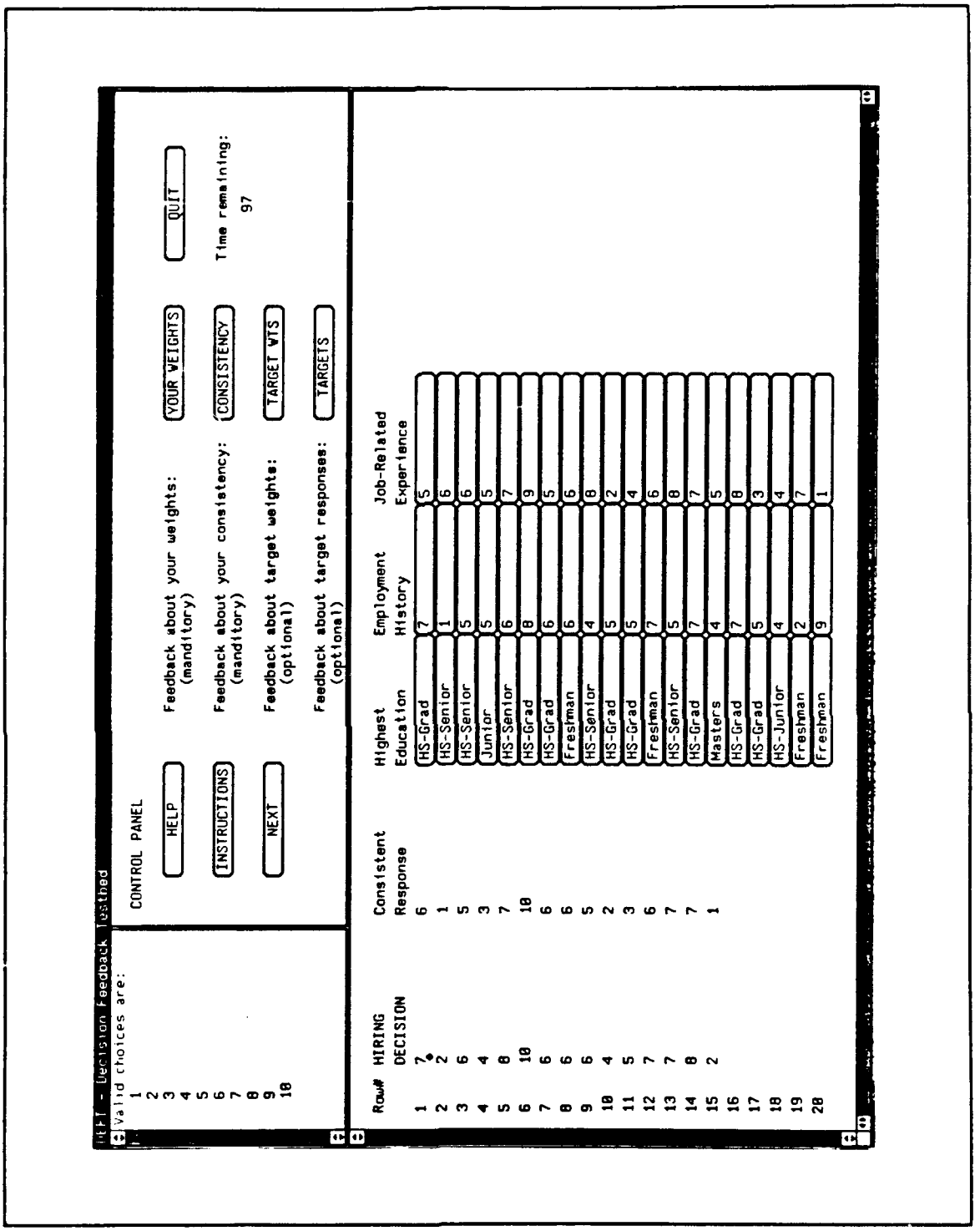

Figure 6. Sample Consistent User Responses Feedback Screen

Two forms of cognitive feedback are shown as bar-graphs, the estimated weights for the ecological validity (TI) and the cue utilization (CI). The user may not be familiar with these terms, so DEFT refers to TI feedback as "target weights" and CI feedback as "user weights". Both are drawn in a similar manner.

The bar-graph weights are calculated from the betas (β_e and β_s) described in Chapter II. The β_e matrix for ecological validity, and the β_s matrix for cue utilization. The weight for each cue is found by dividing the square of its beta by the summation of the square of all betas, then multiplying the result by 100 to convert it to a percent. The sign of the cue's beta determines if the weight is positively or negatively correlated. Figure 7 shows a sample user weights feedback screen. Figure 8 shows a sample target weights feedback screen. The user presses the Continue button when he has completed reviewing the feedback.

H. TEST DATA

There are two major ways to construct test data (Libby, 1981, pp. 39-40). The first is called *representative design*, where test cases are extracted from actual cases, or fabricated cases using realistic relationships between cues and events. The second way to construct test cases is called *systematic design*, where test cases are not representative of actual cases, but are designed for statistical analysis. According to Libby, in the representative design approach most cues correlate with the environment or other cues, and determining the cue utilization estimates is more difficult. DEFT imports test data via the Experiment File's Block command and does not identify or correct for these correlations. The experimenter should create randomized and counterbalanced test data to present accurate feedback.

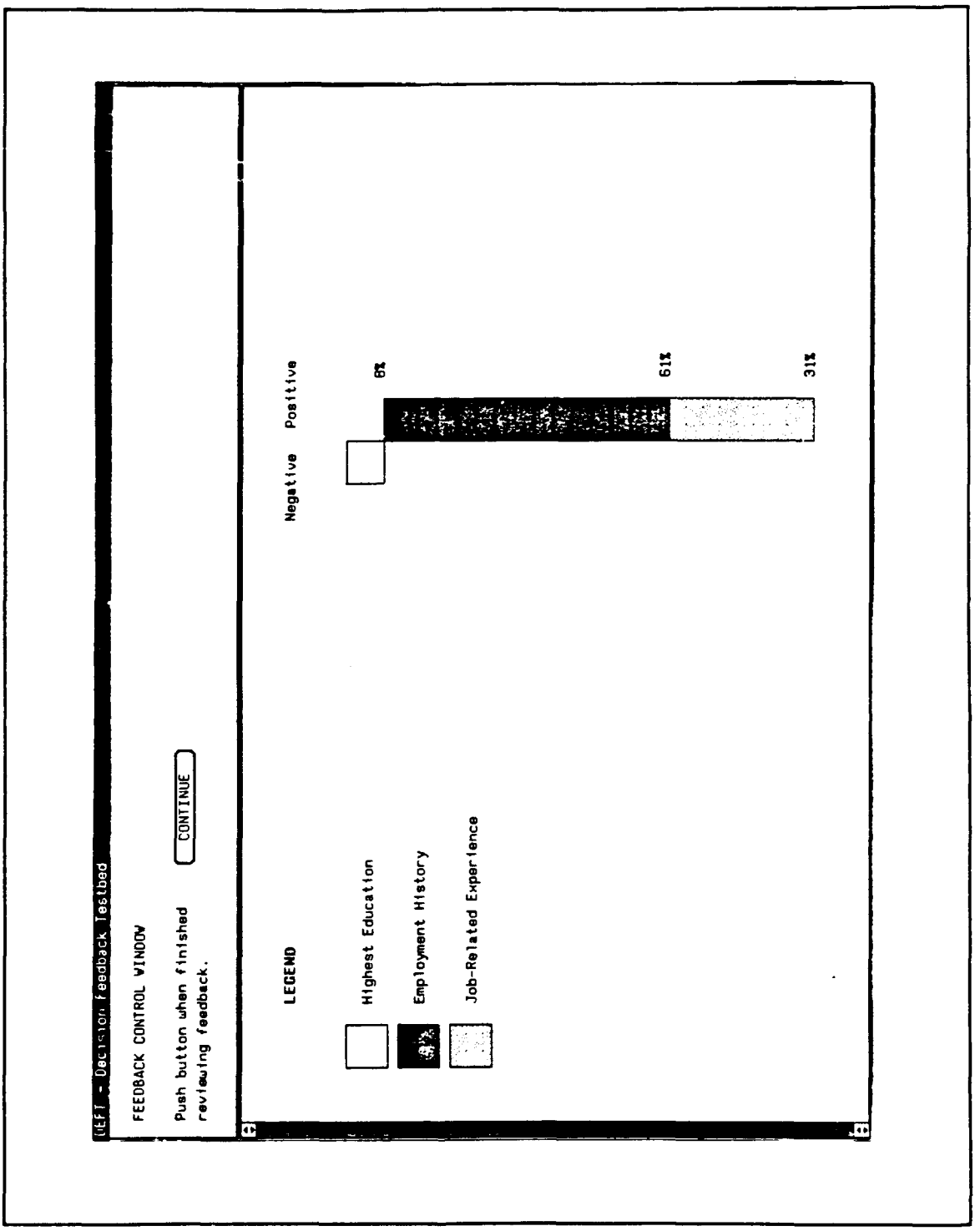

Figure 7. Sample User Weights Feedback Screen

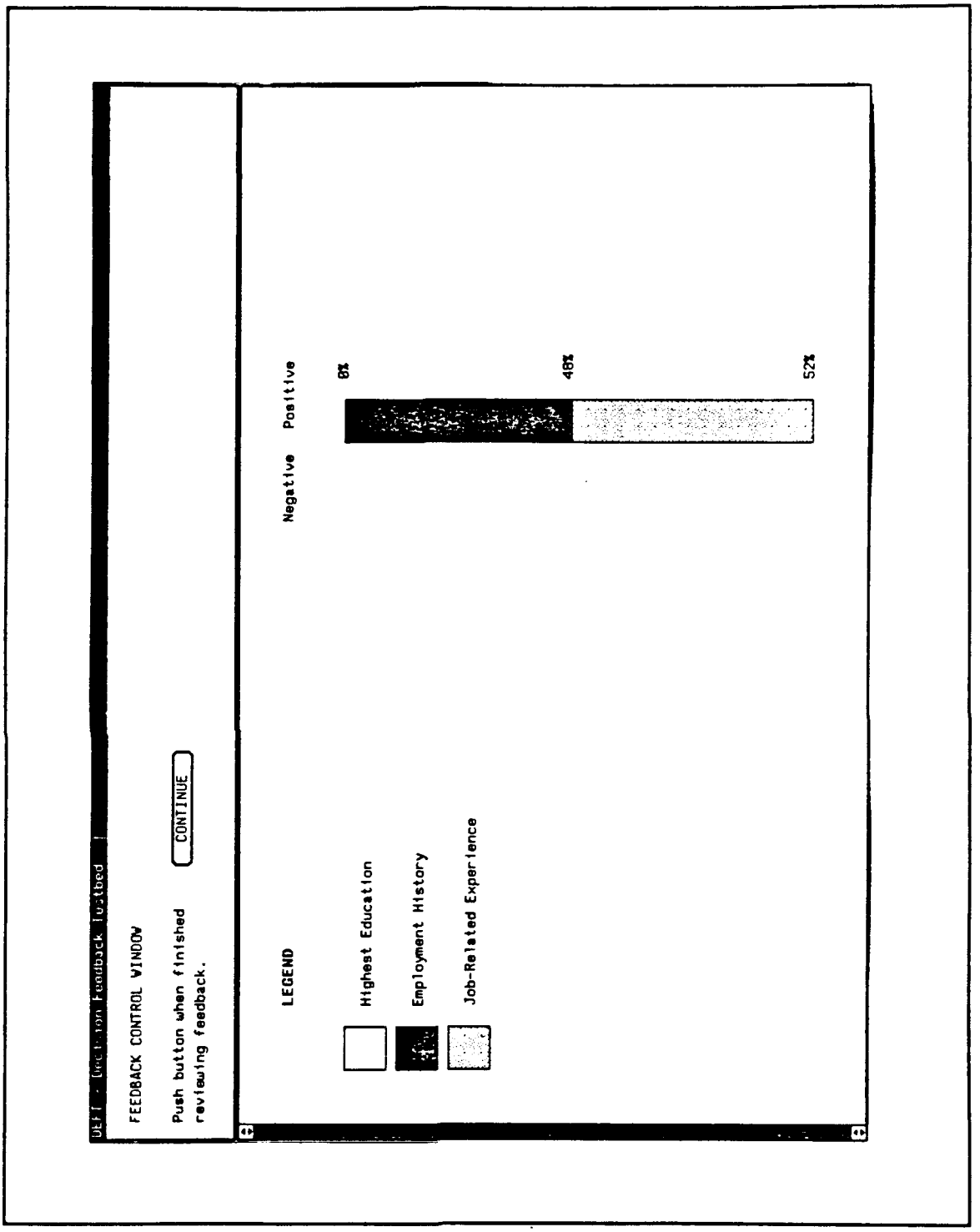

Figure 8. Sample Target Weights Feedback Screen

IV. DEFT FILES

This chapter describes the files used by DEFT. These files are divided into three categories: text, control, and output. DEFT uses files to store text to facilitate revisions and because of their large size. There are three basic types of text files: help, instruction, and questionnaire. Help text is stored in a file named "deft.help". This is the only file used by DEFT with a fixed name. Instruction and questionnaire texts are stored in multiple files, created by the experimenter, under any file names. DEFT is informed of these file names by commands within the Experiment File. Instruction and questionnaire text files are described in additional detail by the Experiment File commands: Instruct, and Question.

The second category of files are control files. The experimenter interacts with experimental subjects (users) through a series of four files types. These file types are: Experiment, Format Set, Format, and Block. The third category of files are output files. DEFT provides the experimenter with experimental results through two output file types: Feedback History, and Iteration History. These files are described in more detail later in this chapter.

A. EXPERIMENT FILE

Before running DEFT, the experimenter must create the experiment configuration files. The first of these files is the Experiment File. This file controls all aspects of the experiment. This includes: instructions to the user, the appearance of test data, the types of feedback shown, time limits, and what questionnaires the user must answer. The Experiment File is an ASCII text file

which contains commands. There are thirteen Experiment File commands. Ten of the commands affect the appearance of the user interface and three affect the output file. Commands can be given more than once and in almost any order. See Table 1 for a summary of these commands and Figure 9 for a sample Experiment File.

TABLE 1. EXPERIMENT FILE COMMANDS

Command	Parameter
Block	Block File Name
Display	All or One
Expcond	decimal number
Formats	Format Set File Name
Group	decimal number
Instruct	Instruction File Name
Phase	decimal number
Question	Questionnaire File Name
Targets	Off, Cond (conditional), or Mand (mandatory)
Timer	decimal number
Usercon	Off, Cond (conditional), or Mand (mandatory)
Userwts	Off, Cond (conditional), or Mand (mandatory)
Weights	Off, Cond (conditional), or Mand (mandatory)

1. Block Command

The Block command identifies the file containing the experimental data set. The command contains one parameter which is the file name. The experimenter stores the block in an ASCII text file which is read and displayed to the user. The format of this file is explained later in this chapter in the Block File section. The appearance of the block to the user is controlled by the Formats command. A Formats command must precede a Block command. In addition, the Block command is a signal to display the block to the user. Other commands, such as Timer, are not executed until a block is displayed.

```
display     all
userwts     mand
usercon     mand
weights     cond
targets     cond
timer       180.0
formats     hire.format.set
block       hire.data.set
```

Figure 9. Sample Experiment File

2. Display Command

The Display command controls how cue values are shown to the user. There are two possible parameters to this command: All or One. If the parameter is All, the user will see all of the cues, all of the time. All is the default value. If the parameter is One, the user must press the cue's button to see

the cue's value. Once a cue's button is pressed, the cue's value remains visible throughout the entire block, even if another button is pressed. DEFT tracks which cues the user has seen and writes this data to an output file. DEFT does not analyze this data to see what the user looks at, or what the user ignores.

3. Expcond Command

The Expcond command sets the experimental condition variable. The command contains one parameter which is a decimal number. This number is written to output files and analyzed by another program. This number does not affect the user interface or program control. The default value is one.

4. Formats Command

The Formats command identifies the file containing the format set. The command contains one parameter which is the file name. The experimenter stores the format set in an ASCII text file. A format set controls the appearance of the block of data shown to the user. The format of this file is explained later in this chapter in the Format Set File section. A Formats command must precede a Block command. Each block of data following a Formats command will use that format set, until another Formats command is executed.

5. Group Command

The Group command sets the within group sequence variable. The command contains one parameter which is a decimal number. This number is written to output files and analyzed by another program. This number does not affect the user interface or program control. The default value is one.

6. Instruct Command

The Instruct command presents instructions to the user. The experimenter writes instructions and stores them in an ASCII text file. Different

instructions are stored in different files. The command contains one parameter which is the file name. Instructions are displayed within a scrolling text window that the user cannot change. The user presses the Continue button when he has completed reading the instructions. In addition, the user can review the latest instructions by pressing the Instruction button.

7. Phase Command

The Phase command sets the phase variable. The command contains one parameter which is a decimal number. This number is written to output files and analyzed by another program. This number does not affect the user interface or program control. The default value is one.

8. Question Command

The Question command presents a questionnaire to the user. The experimenter writes a questionnaire and stores it in an ASCII text file. Different questionnaires are stored in different files. The command contains one parameter which is the file name. Questionnaires are displayed within a scrolling text window that the user edits. The user responds to each question directly within the text window. The user presses the Finished button when he has completed the questionnaire. Completed questionnaires are stored under a file name that consists of the questionnaire file name concatenated with the user's last name. The original questionnaire is not changed.

9. Targets Command

The Targets command controls the display of target responses (criterion) to the user. Target responses are a form of outcome feedback. In this program, target responses represent the correct, or desired, user response. There are three possible parameters to this command: Off, Cond, or Mand. If

the parameter is Off, the user will not see this form of feedback. If the parameter is Cond (conditional), the user can elect to receive this feedback at any point. If the parameter is Mand (mandatory), the user can elect to receive this feedback at any point, but he must receive it at least once. To prevent the target response from becoming the user's response, this feedback is only given when the user has already made a judgment, and the user cannot change a judgment after seeing it. The default value is Off. The Target command is executed when a Block command is executed.

10. Timer Command

The Timer command puts the user under time pressure. The command contains one parameter which is a decimal number. The user is presented with a digital clock counting down the seconds to zero. When time expires, user input is longer accepted. The timer does not affect user feedback. Feedback can be viewed after time elapses, but if time still remains, time spent reviewing feedback counts against the clock. A timer value of zero indicates that there is no time limit. The default is zero. The Timer command is executed when a Block command is executed.

11. Usercon Command

The Usercon command controls the display of consistent user response feedback to the user. Consistent user responses are a form of cognitive feedback. The program analyzes the user's responses, builds a model of the user's cognitive process, and computing a consistent response based on the cues presented. There are three possible parameters to this command: Off, Cond, or Mand. If the parameter is Off, the user will not see this form of feedback. If the parameter is Cond (conditional), the user can elect to receive this feedback at any point. If the

parameter is Mand (mandatory), the user can elect to receive this feedback at any point, but he must receive it at least once. User input is not affected by this feedback, and the user is free to change a judgment after seeing it. The default value is Off. The Usercon command is executed when a Block command is executed.

12. Userwts Command

The Userwts command controls the display of user weight feedback to the user. User weights are a form of cognitive feedback. The program analyzes the user's responses, builds a model of the user's cognitive process, and create a bar graph of the weight the user has assigned to each cue. There are three possible parameters to this command: Off, Cond, or Mand. If the parameter is Off, the user will not see this form of feedback. If the parameter is Cond (conditional), the user can elect to receive this feedback at any point. If the parameter is Mand (mandatory), the user can elect to receive this feedback at any point, but he must receive it at least once. User input is not affected by this feedback, and the user is free to change a judgment after seeing it. The default value is Off. The Userwts command is executed when a Block command is executed.

13. Weights Command

The Weights command controls the display of target weight feedback to the user. Target weights are a form of cognitive feedback. The program analyzes the target responses and create a bar graph of the weight assigned to each cue. There are three possible parameters to this command: Off, Cond, or Mand. If the parameter is Off, the user will not see this form of feedback. If the parameter is Cond (conditional), the user can elect to receive this feedback at

any point. If the parameter is Mand (mandatory), the user can elect to receive this feedback at any point, but he must receive it at least once. User input is not affected by this feedback, and the user is free to change a judgment after seeing it. The default value is Off. The Weights command is executed when a Block command is executed.

B. FORMAT SET FILE

Formats controls the appearance of the cues shown to the user, and each cue has a format. A cue can be numerical, discrete, or categorical. In addition to cues, the target response has a format. If the target response is numerical, the user will have to enter a number within a given range. Formats are identified by a file name and they are described in the next section of this chapter (Format File). The Format Set File is an ASCII text file which lists all of the formats.

The first file entry is an integer that equals the number of cues to be shown to the user plus one. The second file entry is the format identifier of the criterion/target response. The third file entry is the format identifier of the first cue. The entries continue with a format identifier for each cue. The number at the beginning of the file should equal the number of format identifiers listed. See Figure 10 for a sample Format Set File.

```
4
hire.format
education.format
employment.history.format
experience.format
```

Figure 10. Sample Format Set File

C. FORMAT FILE

A Format File controls the appearance of a cue. The first file entry is the first line of the cue's title. The second file entry is the second line of the cue's title. Each title can be 14 characters in length, but spaces are not allowed. The third file entry is the format type. There are three types of Format Files: N (numerical), D (discrete), or C (categorical). Each format type requires different follow-on entries.

1. Numerical Format Type

The numerical format type limits a cue or a response to a range of numbers. This type is continuous, meaning it can take on any value within this range. There are three follow-on file entries associated with numerical types. The first entry is the display decimal precision, and it is an integer. This does not limit the cue or response value, only how that value is shown to the user. A display decimal precision of zero means all cues or responses are shown as integers. The second entry is the lowest allowable value, and it is a decimal number. The final entry is the highest allowable value, and it is a decimal number.

2. Discrete Format Type

The discrete format type limits a cue, or a response, to a set of given choices. For a discrete type, follow-on entries in the Format File consist of an unlimited number of discrete pairs. Here are three examples:

- Integer {(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7)}
- Years in School {(Freshman, 0), (Sophomore, 1), (Junior, 2), (Senior, 3)}
- Liquid Measure {(Cup, 1), (Pint, 2), (Quart, 4), (Gallon, 16)}

While a discrete type implies a specific number, what is shown to the user may be more complex. A discrete pair is a display string, with a maximum of 14 characters, and an associated decimal number representing its value. See Figure 11 for a sample Format File of a discrete type.

```
Student
Grade

d

A           4.0
A-          3.7
B+          3.3
B           3.0
B-          2.7
C+          2.3
C           2.0
C-          1.7
D+          1.3
D           1.0
D-          0.7
F           0.0
```

Figure 11. Sample Format File for a Discrete Type

3. Categorical Format Type

The categorical format type limits cues and responses to given categories. While categories have a central relationship, they do not have a numerical value. This does not mean that categories cannot be ordered. The price of a home may be given as high, medium, or low. The home price is not discrete because a precise numerical value cannot be associated with each of these

categories. Their "fuzziness" makes them inappropriate as a discrete type. Here are three examples:

- Store Location {(Chicago, 1), (Miami, 2), (Seattle, 3), (Houston, 4)}
- Student Sex {(Female, 0), (Male, 1)}
- Car Color {(Red, 1), (Blue, 2), (Yellow, 3), (Green, 4)}

To simplify data entry, each category is given an associated number. This number is used in the Block File to refer to a category, but it has no computational significance. For a categorical type, follow-on entries in the Format File consist of an unlimited number of categorical pairs. A categorical pair is a display string, with a maximum of 14 characters, and an associated decimal number. When creating a cognitive model of a system with a categorical cue, each category is treated as a separate cue. A binary value of one is assigned to the category that matches the cue, and all others are assigned a value of zero (Goldberg, 1986). Since each category is actually a separate cue, and since the underlying model in DEFT can only handle a single dependent response, categorical formats cannot be used as the criterion/target response.

D. BLOCK FILE

The Block File contains a set of target responses and cues. Each target response, followed by its associated cues, makes one "task". The first file entry is the number of tasks in the block or data set, and it is an integer. The second entry is the minimum number of user responses before the user can receive cognitive feedback, and it is an integer. This number is closely related to the number of cues and categories. The follow-on entries in the Block File consist of an unlimited number of tasks. The first value in a task must be the

criterion/target response, followed by its associated cues. These values are decimal numbers. See Figure 12 for a sample Block File.

```
20
12

5       3       7.38071     5
2       2       1.00000     6
5       2       5.31848     6
4       6       4.66804     5
6       2       6.45272     7
7       3       7.95032     9
4       3       6.44342     5
6       4       6.38042     6
5       2       3.81946     8
2       3       5.00729     2
4       3       5.00223     4
6       4       6.61737     6
6       2       5.32304     8
7       3       6.84553     7
4       9       4.41550     5
8       3       7.38217     8
2       3       4.79100     3
3       1       3.91903     4
3       4       1.62861     7
4       4       9.00000     1
```

Figure 12. Sample Block File

E. FEEDBACK HISTORY FILE

DEFT asks the user to enter judgments (user responses) to a set of cues. This series of "tasks" makes up one "block". DEFT then provides the user with cognitive feedback to test its effect on user training and task performance. The

user does not have to complete the entire block before requesting this feedback. DEFT requires a minimum number of judgments to analyze before it will provide feedback, but the user can request feedback anytime after providing this minimum number. To determine how the user changes his judgments after feedback, DEFT records each "iteration" of the block. The Feedback History File records which types of feedback the user reviewed during each iteration.

For each iteration, DEFT writes one line in the Feedback History File. This line contains six items: block number, iteration number, number of times the user received user weight feedback, number of times the user received user consistency feedback, number of times the user received target weight feedback, and number of times the user received target response feedback.

When the user complete the experiment, two last lines are written in the Feedback History File. The first of these lines contains four items: first name, last name, age, and sex. The final line contains six items: experimental condition variable, student mail box code (SMC), experimental within group sequence variable, phase variable, and two zeros. DEFT writes these last two zeros to match the input needs of another program. The Feedback History File is stored under a file name that consists of the Experiment File name concatenated with the user's last name. See Figure 13 for a sample Feedback History File.

F. ITERATION HISTORY FILE

To determine how users change their judgments after feedback, DEFT records each iteration of the block. The first entry in the Iteration History File is the number of tasks, or rows, in the block. This number is not the number of user responses, but the total number of tasks in the block.

```
1  1  2  2  1  1
1  2  1  1  0  0
1  3  1  1  0  1
2  1  1  2  1  0
2  2  1  1  0  0

John Smith 25 M

1  1979  1  1  0  0
```

Figure 13. Sample Feedback History File

For each task in the block, DEFT writes one line in the Iteration History File. This line contains the set of cue pairs followed by the criterion, the user's response, and the status of the user's response.

The set of cue pairs consists of one cue pair for each cue in the task. A cue pair contains the cue value and cue display flag. The cue display flag is an integer. If it is zero, the user has not seen the cue. If it is a positive integer, the flag represents the order that the user displayed the cue. There is one exception, if the Display command is set to All, the cue display flag is always one.

The status of the user's response can take three values: zero (if the user has not entered a judgment), one (if the user has entered an invalid judgment), or two (if the user has entered a valid judgment).

The last four lines in the Iteration History File have a fixed format. The first of these lines contains five items: number of keystrokes entered, number of times the up-arrow key was pressed, number of times the down-arrow key was pressed, number of times the carriage return key was pressed, and number of

keystrokes made in error. DEFT does not track these items, and they are always zero. DEFT writes these items to match the input needs of another program.

The third to last line consists of the elapsed time, in seconds, for the iteration. The second to last line contains four items: first name, last name, age, and sex. The final line contains six items: experimental condition variable, student mail box code (SMC), experimental within group sequence variable, phase variable, block number, and iteration number.

The Iteration History Files is stored under a file name that consists of the Block File name concatenated with the user's last name, block number, and iteration number. See Figure 14 for a sample Iteration History File.

```
20

3 1     7.38071 1     5 1     5     5 2
2 1     1.00000 1     6 1     2     3 2
2 1     5.31848 1     6 1     5     4 2
6 1     4.66804 1     5 1     4     5 2
2 1     6.45272 1     7 1     6     5 2
3 1     7.95032 1     9 1     7     6 2
3 1     6.44342 1     5 1     4     5 2
4 1     6.38042 1     6 1     6     5 2
2 1     3.81946 1     8 1     5     4 2
3 1     5.00729 1     2 1     2     3 2
3 1     5.00223 1     4 1     4     4 2
4 1     6.61737 1     6 1     6     5 2
2 1     5.32304 1     8 1     6     5 2
3 1     6.84553 1     7 1     7     5 2
9 1     4.41550 1     5 1     4     0 0
3 1     7.38217 1     8 1     8     0 0
3 1     4.79100 1     3 1     2     0 0
1 1     3.91903 1     4 1     3     0 0
4 1     1.62861 1     7 1     3     0 0
4 1     9.00000 1     1 1     4     0 0

94

John Smith 25 M

1 1979 1 1 1 1
```

Figure 14. Sample Iteration History File

V. CONCLUSION

A. SUMMARY

Decision making is a topic of great interest to many groups and organizations. Although decision making has occured for as long as people have been around, scientific theories that could be empirically tested were only developed in the last 50 years. Within the Department of Defense there is a constant need to improve decision making in repetitive situations. Decision researchers have described many new methods to study and improve decision making behavior. One method of interest is the use of feedback.

Feedback is an adaptive method. People change their decision making process based on new information they receive about their performance. To identify the effects of feedback, one must be able to model both the environment and the decision maker. Cognitive feedback theories have been shown to improve decision making performance, but there have been few empirical studies which identify the types, and forms of cognitive feedback that are most effective. This study designed and built a testbed to evaluate cognitive feedback theories.

This testbed is a computer program which infers decision strategies, provides feedback, and monitors the effectiveness of the feedback. The program uses a computer mouse and information board to implement process tracing based on the information acquisition search methodology.

B. CONCLUSION

This study had two major objectives. The first objective was to review decision and cognitive feedback theory, determine their theoretical basis, then

identify areas which lend themselves to computer automation. The second objective was to develop a platform upon which to test cognitive feedback theories. Both of these objectives were met.

Chapter II answered the question of how to monitor the decision maker's information acquisition processes and the effects of feedback on them. Brunswik's lens model was shown to represent the decision making process in repetitive decision situations. Simple linear regression was shown to successfully model the decision maker's cognitive processes, and outcome and cognitive feedback were explained. Finally, the information acquisition search technique was discussed as a process tracing methodology to infer decision strategies.

Chapters III and IV answered the question of how to design and write a program to test and monitor the decision making processes and the effects of feedback. The program called DEFT, *DEcision Feedback Testbed*, was written using structured programming techniques. Coding and integration testing of DEFT is finished.

C. RECOMMENDATIONS

There are four recommendations. The first recommendation is to make DEFT the subject of a follow-on study. DEFT has not been used by novices and its effectiveness has not evaluated. The second recommendation is to expand DEFT to include other forms of feedback. This should include new types of feedback (e.g., the matching index between the predicted decision maker's response and the predicted criterion event) and presenting the feedback in new formats (e.g., side-by-side graphs and tables). The third recommendation is to conduct further research to determine what domains or tasks within the Department of Defense would benefit from this tool. The final recommendation

is to apply DEFT to these new domains and task and empirically test the effects of cognitive feedback on decision maker performance.

LIST OF REFERENCES

[Arkes and Hammond, 1986] Arkes, H. R. and Hammond, K. R., *Judgment and Decision Making*, Cambridge: Cambridge University Press, pp. 6-8, 1986.

[Balzer, Doherty, and O'Conner, 1989] Balzer, W. K., Doherty, M. E., and O'Conner, R., "Effects of Cognitive Feedback on Performance" in *Psychological Bulletin*, v. 106, n. 3, pp. 410-427, 1989.

[Berry, 1987] Berry, D.C., "The Problem of Implicit Knowledge" in *Expert Systems*, v. 4, pp. 144-151, 1987.

[Brunswik, 1955] Brunswik, E., "The Conceptual Framework of Psychology" in *International Encyclopedia of Unified Sciences*, ed. Neurath, O., Carnap, R., and Morris, C., v. 1, n. 6, Chicago: University of Chicago Press, pp. 655-760, 1955.

[Cooksey, Freebody, and Davidson, 1986] Cooksey, R. W., Freebody, P., and Davidson, G. R., "Social Judgment Theory: Teacher Expectations Concerning Children's Early Reading Potential" in *Judgment and Decision Making*, ed. Arkes, H. R. and Hammond, K. R., Cambridge: Cambridge University Press, pp. 527-528, 1986.

[Doherty and Balzer, 1988] Doherty, M. E. and Balzer, W. K., "Cognitive Feedback" in *Human Judgement: The SJT View*, ed. Brehmer, B. and Joyce, C. R. B., North-Holland, pp. 163-164, 1988.

[Dudycha and Naylor, 1966] Dudycha, A. L. and Naylor, J. C., "Characteristics of the Human Inference Process in Complex Choice Behavior Situations" in *Organizational Behavior and Human Performance*, v. 1, pp. 110-128, 1966.

[Ericson and Simon, 1984] Ericson, K. A. and Simon, H. A., "Verbal Reports as Data", Cambridge: MIT Press, 1984, quoted in Johnson, E. J., Payne, J. W., Schkade, D. A., and Bettman, J. R., *Monitoring Information Processing and Decisions: The Mouselab System*, pp. 1-5, 1989.

[Goldberg, 1986] Goldberg, L. R., "Simple Models or Simple Processes? Some Research on Clinical Judgments" in *Judgment and Decision Making*, ed. Arkes, H. R. and Hammond, K. R., Cambridge: Cambridge University Press, pp. 338-348, 1986.

[Hartson and Hix, 1988] Hartson, H. R. and Hix, D., "Toward Empirically Derived Methodologies and Tools for Human-Computer Interface Development" in *International Journal of Man - Machine Studies*, v. 31, pp. 477-494,1989.

[Hartson and Hix, 1989] Hartson, H. R. and Hix, D., "Human-Computer Interface Development: Concepts and Systems for its Management" in *ACM Human-Computer Interface Development*, pp. 1-11,1989.

[Johnson, Payne, Schkade, and Bettman, 1989] Johnson, E. J., Payne, J. W., Schkade, D. A., and Bettman, J. R., *Monitoring Information Processing and Decisions: The Mouselab System*, pp. 1-5, 1989.

[Libby, 1981] Libby, R., *Accounting and Human Information Processing: Theory and Applications*, Englewood Cliffs, NJ: Prentice-Hall, pp. 18-40, 1981.

[Neisser, 1967] Neisser, U., *Cognitive Psychology*, New York: Meredith Publishing, pp. 3-4, 1967.

[Olson and Rueter, 1987] Olson, J. R. and Rueter, H. H., "Extracting Expertise From Experts: Methods of Knowledge Acquisition" in *Expert Systems*, pp. 152-168, 1987.

[Payne, Braunstein, and Carroll, 1978] Payne, J. W., Braunstein, M. L., and Carroll, J. S., "Exploring Predecisional Behavior: An Alternative Approach to Decision Research" in *Organizational Behavior and Human Performance*, n. 22, pp. 17-44, 1978, quoted in Johnson, E. J., Payne, J. W., Schkade, D. A., and Bettman, J. R., *Monitoring Information Processing and Decisions: The Mouselab System*, p. 2, 1989.

[Powers, Cheney, and Crow, 1990] Powers, M. J., Cheney, P. H., and Crow, G., *Structured Systems Development - Analysis, Design, Implementation*, Boston: Boyd & Fraser, pp. 454-464, 1990.

[Russo, 1978] Russo, J. E., "Eye Fixations Can Save the World: A Critical Evaluation and Comparison Between Eye Fixations and Other Information Processing Methodologies" in *Advances in Consumer Research*, ed. Hunt, H. K., v. V, pp. 561-570, 1978, quoted in Johnson, E. J., Payne, J. W., Schkade, D. A., and Bettman, J. R., *Monitoring Information Processing and Decisions: The Mouselab System*, pp. 1-5, 1989.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2. Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3. Commandant (G-PO-2) United States Coast Guard 2100 Second Street, SW Washington, District of Columbia 20593-0001	2
4. Professor Kishore Sengupta, Code 54SE Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5004	1
5. Professor Tung X. Bui, Code 54BD Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5004	1
6. LT Paul Kevin Larson 504 Waverly Avenue Streamwood, Illinois 60107	2